Keyword Extraction with Table based KNN Variants

Taeho Jo
President
Alpha AI Research
Cheongju, South Korea
tjo018@naver.com

***Abstract*—In this research, we propose and apply the table based KNN variants to the keyword extraction. The initial KNN version was previously modified into the table-based version and applied by mapping the keyword extraction into a binary classification. In this research, we mentioned the three KNN variants, in case of the numerical vector-based versions: one where the selected nearest neighbors are discriminated by their similarities, one where the attributes are discriminated by their correlations with the categories, and one where the training examples are discriminated by their credits. In this research, the three KNN variants are modified into the table-based versions, as well as the initial KNN version. The goal of this research is to improve the keyword extraction performance by modifying them so.**

I. Introduction

The keyword extraction refers to the process of selecting and extracting important words from a text. It was interpreted into a binary word classification where each word is classified into keyword or non-keyword, in previous works. The keyword extraction is regarded as a domain dependent task where a same word may be classified differently, depending on the given domain, differently from the topic-based word classification where each word is classified into one or some among the predefined topics. In the keyword extraction system, a text is indexed into a list of words, each word is classified into keyword or non-keyword by a machine learning algorithm, and words which are classified into keyword are extracted, externally. This research scope is restricted to the crisp keyword extraction where each word is classified exclusively into one of the two categories, and it will be expanded into the fuzzy keyword extraction or the hierarchical keyword extraction.

This research is motivated by the successful results in applying the table based KNN algorithm to the keyword extraction in the previous work. Another motivation is the successfulness in using the KNN version for the topic-based word classification, as well as the keyword extraction. Words were encoded into tables as the alternative representations to the numerical vectors, and the KNN algorithm was modified into the table-based version as the approach to the word categorization and the keyword extraction. Its better performance was empirically validated in both tasks, compared with the traditional KNN algorithm. In this research, we derive the KNN variants and modify them into the table base versions as the approaches to the keyword extraction.

The idea of this research is to modify the three KNN variants into the table-based versions which process tables directly and use them for the keyword extraction. In the first variant, nearest neighbors are discriminated by their similarities with or distances from a novice example, in voting their labels. In the second variant, the table entries are discriminated by their correlations with the output values in computing similarities of a novice example with the training examples. In the last variant, the training examples are discriminated by their qualities in computing their similarities with a novice item or voting nearest neighbors. We modify the three KNN variants into the table-based version, in order to improve their performances.

Let us mention some benefits from this research. Words are represented into alternative structured forms, tables, for solving the problems in encoding words into numerical vectors. The KNN algorithm is replaced with the KNN variants for improving the classification performance. The KNN variants are modified into the table-based versions for improving the classification performance, additionally. The better tools than the KNN algorithms are provided for implementing the keyword extraction system by this research.

Let us mention the organization of this paper. In Section II, we explore the previous works which are relevant to this study. In Section III, we describe in detail what is proposed in this research. In Section IV, we mention the significances of this research and the remaining tasks. This paper is composed of the four sections.

II. Previous Works

This section is concerned with the previous works which are relevant to this research. It is intended to expand the style of modifying the KNN algorithm to its three variants as approaches to keyword extraction. The directions of exploring previous works are derivation of KNN variants, modification of the standard KNN algorithm into its table-based version applied to the keyword extraction, and the previous case of encoding a text into a table. The distinguishable points of this research are derivation of three KNN variants from the standard version, modification of them into their table-based versions, and application of them to the keyword extraction This article is intended to explore previous works for providing the background for this research.

Let us explore the previous works on KNN variants. In 2001, the KNN variant where nearest neighbors are discriminated by their distances from or similarities as a novice item was applied to the text classification by Han et al. [1]. In 2008, MKNN (Modified K Nearest Neighbor) the KNN variants where training examples are discriminated by their validness, was proposed by Parvin et al. [3]. In 2018, the KNN variant where the attributes are discriminated by their correlations were proposed by Nababan and Sitompul [9]. However, this research uses different specific metrics for discriminating attributes, nearest neighbors, and training examples.

Let us explore the previous works on applying the modified version of KNN algorithm to the keyword extraction. In 2016, it was initially proposed that the table-based version of KNN algorithm should be applied to the keyword extraction by Jo [6]. In 2018, the modified KNN algorithm was validated in the task by Jo [8]. The journal article which describes in detail the modified KNN algorithm and its application to the keyword extraction is finally prepared for its publication by Jo [10]. This research is based on the three previous works.

Let us explore the previous works on the table matching algorithm as the early case of encoding a text into a table. In 2008, the table based matching was proposed as an approach to the text classification as the initial case of encoding a text into a table, instead of a numerical vector [2]. In 2011, the table based matching algorithm was registered as a patent by Jo [4]. In 2015, it was upgraded into its more reliable and stable version by Jo [5]. The similarity metric between two tables is provided by the above previous works for modifying the standard KNN algorithm and its variants.

Let us mention the differences of this research from the previous works which were explored above. The KNN variants in the previous works are numerical vector-based versions, and in this research, they will be modified into table-based versions in this research. As the expansion of [10], we modify and adopt the KNN variants for implementing the keyword extraction system. The modified versions of KNN variants become more advanced ones, compared with the table based matching algorithm. The modified KNN variants will be described in detail in Section III.

III. Proposed Work

This section is concerned with what is proposed in this research. In Section III-A, we describe the process of encoding a word into a table and the proposed similarity metric. In Section III-B, we describe the standard version of KNN algorithm where the proposed similarity metric is adopted. In Section III-C, we derive the three varaints from the standard version. In Section III-D, we present the system architecture of the keyword extraction system.

A. Similarity Metric

This section is concerned with the table as the representation of a word. The table is defined as a set of entries, each of which consists of an element and its weight. The table which represents a word consists of entries each of which contains a text identifier and its weight. The similarity between tables is computed based on shared words as the semantic similarity between words. This section is intended to describe the process of encoding a word into a table, and the process of computing the similarity between tables.

The process of encoding words into tables is illustrated in Figure 1. In the inverted indexing, each word is linked with texts which include itself. In the text-word weighting, its weights in the lined texts are computed; a weight indicates a relationship between a word and a text. Entries with their lower weights are removed for downsizing the table. Refer to [7] for studying the encoding process in more detail.

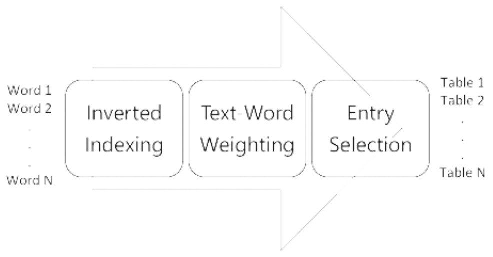

Figure 1. Process of Encoding Words into Tables

Let us mention the function of a table which maps it into a set of text identifiers as shown in Figure 2. The table which represents a word is expressed as entry set as shown in equation (1),

$$T = \{(d_1, w_1), (d_2, w_2), \ldots, (d_{|T|}, w_{|T|})\} \tag{1}$$

where d_i is a text identifier and w_i is the weight of the word, T, in the text, d_i. The function for generating a list of text identifiers is expressed as equation (2),

$$F(T) = \{d_1, d_2, \ldots, d_{|T|}\} \tag{2}$$

The function is the operation which takes only text identifiers without their weights as a set. The operation is applied to computing the similarity between tables which represent words.

Let us mention the process of computing the similarity between two tables as a semantic similarity between words. The two tables which represent words are notated respectively by $T_1 = \{(d_{11}, w_{11}), (d_{12}, w_{12}), \ldots, (d_{1|T|}, w_{1|T|})\}$ and $T_2 = \{(d_{21}, w_{21}), (d_{22}, w_{22}), \ldots, (d_{2|T|}, w_{2|T|})\}$. The

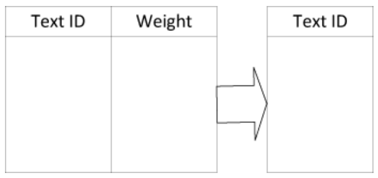

Figure 2. Conversion of Table into Text Set

intersection between the two sets which is generated by applying the function, $F(\cdot)$ as shown in equation (3),

$$F(T_1) \cap F(T_2) = \{d_{s1}, d_{s2}, \ldots, d_{s|F(T_1)\cap F(T_2)|}\} \tag{3}$$

and the table with entries each of which consists of a shared text identifier, d_{si}, its weight from the table, T_1, w^1_{si}, and its weight from the table, T_2, w^2_{si}, is expressed as equation (4),

$$\begin{aligned} ST_{12} = \{&(d_{s1}, w^1_{s1}, w^2_{s1}), (d_{s2}, w^1_{s2}, w^2_{s2}), \ldots, \\ &(d_{s|F(T_1)\cap F(T_2)|}, w^1_{s|F(T_1)\cap F(T_2)|}, w^2_{s|F(T_1)\cap F(T_2)|})\} \end{aligned} \tag{4}$$

The similarity between tables, T_1 and T_2 is computed by equation (5),

$$sim(T_1, T_2) = \frac{2(\sum_{i=1}^{|F(T_1)\cap F(T_2)|} w^1_{si} + \sum_{i=1}^{|F(T_1)\cap F(T_2)|} w^2_{si})}{\sum_{i=1}^{|T_1|} w_{1i} + \sum_{i=1}^{|T_1|} w_{2i}}. \tag{5}$$

The value which is computed from equation (5) is always given as a normalized value between zero and one as shown in equation (6),

$$0 \leq sim(T_1, T_2) \leq 1. \tag{6}$$

Let us make some remarks on the encoding process and the computation of similarity between words. A word is encoded into table by the inverted indexing, the text identifier weighting, and the entry selection. A table is expressed as a set of entries and filtered into a set of text identifiers by the function. The similarity between tables is computed based on their shared entries by equation (5). In the future research, we consider representing a word into multiple tables.

B. *Table based KNN Algorithm*

This section is concerned with the initial version of table based KNN algorithm. It was initially proposed as the approach to the word classification by Jo in 2018 [7]. The similarity metric between tables which was described in the previous section is used for computing the similarity between a novice table and a training example. The labels of its nearest neighbors are voted with their identical weights for deciding the label of a novice input. This section is intended to describe the initial version of KNN algorithm from which its variants are derived.

The process of computing the similarity between a novice item and a training example is illustrated in Figure 3. The labeled words which are gathered as samples are encoded into tables, and they are notated by a set $Tr = \{T_1, T_2, \ldots, T_N\}$. A novice word is encoded into a table notated by T. Its similarities with the tables which represent the sample words are computed by equation (5), as $sim(T, T_1), sim(T, T_2), \ldots, sim(T, T_N)$. Some training examples which are most similar as the novice input are selected as its nearest neighbors.

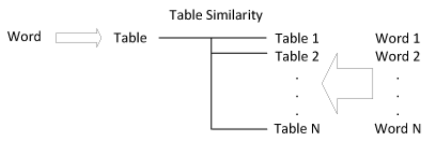

Figure 3. Similarities of Novice Input with Training Examples

In Figure 4, the process of selecting nearest neighbors by the ranking selection is illustrated. The training examples are sorted by the descending order of their similarities, after computing their similarities with a novice example. The training examples with their higher similarities with a novice example are selected as its nearest neighbors, as expressed in equation (7),

$$Nr = Select_k(Tr, T), Nr \subseteq Tr \tag{7}$$

The set of nearest neighbors, Nr, is composed with elements as expressed in equation (8),

$$Nr = \{T^{near}_1, T^{near}_2, \ldots, T^{near}_k\} \tag{8}$$

The elements in the set, Nr, are used for voting their labels in the next step.

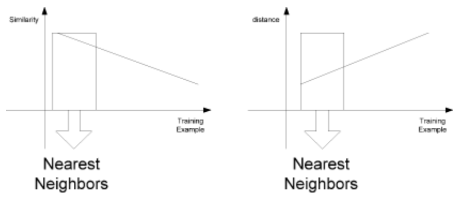

Figure 4. Selection of Most Similar or Least Distant Training Examples as Nearest Neighbors

The process of voting the labels of nearest neighbors for decoding the label of a novice word is illustrated in Figure 5. The predefined categories are notated by $c_1, c_2, \ldots, c_m$, and the label of a nearest neighbor is notated by $c_j = label(T^{near}_i)$. The number of nearest neighbors which belong to the category, c_j, is notated by $Count(Nr, c_j)$. The label of the novice item, T, is decided by equation (9),

$$label(T) = \operatorname*{argmax}_{j=1}^{m} Count(Nr, c_j) \tag{9}$$

The process of deciding the label of a novice item by equation (9), is called voting.

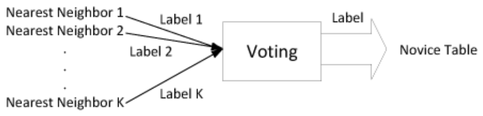

Figure 5. Voting of Labels of Nearest Neighbors for Deciding Label of Novice Input

Let us make some remarks on the initial version of KNN algorithm from which we derive some variants. It computes the similarities of a novice item with the training examples. The training examples with their highest similarities with the novice item are selected as its nearest neighbors. The label of the novice item is decided by voting the labels of its nearest neighbors. The ranking selection is adopted for selecting the nearest neighbors from training examples, in this version.

C. KNN Variants

This section is concerned with the variants which are derived from the KNN algorithm which was covered in Section III-B. The difference from the traditional version is to adopt similarity metric between tables for computing the similarity between a novice item and a training example. In this section, we derive the three variants by discriminating nearest neighbors, entries, or training examples. The three variants of KNN algorithm are adopted for implementing the keyword extraction system. This section is intended to describe the three variants.

Let us mention the KNN variant which is derived by discriminating the nearest neighbors. A set of nearest neighbors is notated by $Nr = \{T_1^{near}, T_2^{near}, \ldots, T_k^{near}\}$, and its elements are assumed as $sim(T, T_1^{near}) \geq sim(T, T_2^{near}) \geq \ldots \geq sim(T, T_k^{near})$. The weights, $w_1, w_2, \ldots, w_k$, are assigned to the nearest neighbors, $T_1^{near}, T_2^{near}, \ldots, T_k^{near}$, following, $w_1 \geq w_2 \geq \ldots \geq w_k$, and the total weights are computed for the category, c_j by equation (10),

$$CS(Nr, c_j) = \sum_{T_i \in Nr} w_i \tag{10}$$

The label of the novice item, T, is decided by equation (11),

$$label(T) = \operatorname*{argmax}_{j=1}^{m} CS(Nr, c_j) \tag{11}$$

There are two ways of assigning weights to the nearest neighbors: exponentially decreasing way and arithmetic decreasing way as shown in Figure 6.

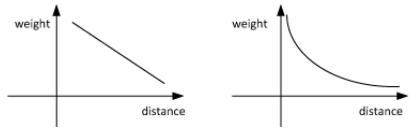

Figure 6. Discrimination over Nearest Neighbors

Let us mention the second table based KNN variant where the entries are discriminated for computing the similarity between a novice item and a training example as shown in Figure 7. The occurrences of the text, d_i, in the words which are labeled with the category, c_j, is $f(d_i|c_j)$, and the total occurrences of the text, d_i in the sample words is $f(d_i)$. The influence of the text, d_i, on the category, c_j, is computed by equation (12),

$$I(d_i, c_j) = \frac{f(d_i|c_j)}{f(d_i)} \tag{12}$$

and the weight, w_i, is computed by equation (13),

$$w_i = \max_{j=1}^{m} I(d_i, c_j). \tag{13}$$

Equation (5) for computing the similarity between tables is modified into equation (14),

$$sim(T_1, T_2) = \frac{2(\sum_{i=1}^{|F(T_1) \cap F(T_2)|} w_i w_{si}^1 + \sum_{i=1}^{|F(T_1) \cap F(T_2)|} w_i w_{si}^2)}{\sum_{i=1}^{|T_1|} w_i w_{1i} + \sum_{i=1}^{|T_1|} w_i w_{2i}}. \tag{14}$$

In this variant, equation (5) is replaced by equation (14) for computing the similarity between a novice item and a training example.

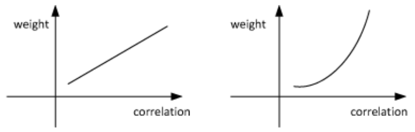

Figure 7. Discrimination over Table Entries

Let us mention the third KNN variant where the training examples are discriminated for computing the similarity between a novice item and a training example and/or voting the nearest neighbors for deciding the label as shown in Figure 8.. The similarity threshold is used as the parameter, and for each training example, other training examples are taken within the threshold from it as its nearest neighbors. The number of nearest neighbors and the number of training examples which are labeled identically to the training example, T_i, are notated respectively by r_i and $r_i^{=}$, and the weight is computed by equation (15),

$$w_i = \frac{r_i^{=}}{r_i} \tag{15}$$

The similarity between the novice item, T, and the training example, T_i, is computed by equation (16),

$$sim(T_1, T_2) = w_i \left(\frac{2(\sum_{i=1}^{|F(T_1) \cap F(T_2)|} w_{si}^1 + \sum_{i=1}^{|F(T_1) \cap F(T_2)|} w_{si}^2)}{\sum_{i=1}^{|T_1|} w_{1i} + \sum_{i=1}^{|T_1|} w_{2i}} \right), \tag{16}$$

and the total weight is computed for the category, c_j, by equation (17), in voting,

$$CS(Nr, c_j) = \sum_{T_i \in Nr} w_i. \tag{17}$$

Equation (16) and (17) are used respectively for computing the similarity between a novice item and a training example and voting the nearest neighbors for each category.

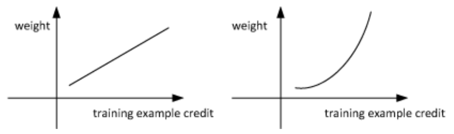

Figure 8. Discrimination over Training Examples

Let us make some remarks on the three variants of KNN algorithm which are proposed as the approaches to the keyword extraction. In the first variant, the nearest neighbors are discriminated for voting their labels. In the second variant, the entries are discriminated for computing the similarity between a novice input and a training example. In the third variant, the training examples are discriminated for voting the labels of the nearest neighbors and/or computing the similarity between tables. We may consider the trainable KNN algorithm which optimizes the weights of the nearest neighbors, the attributes, and the training examples for minimizing the training error.

D. System Architecture

This section is concerned with the architecture and the execution flow of the keyword extraction system. In Section III-C, we described the three KNN variants which are adopted for implementing the keyword extraction system. The initial version of KNN algorithm which is mentioned in Section III-B is replaced by its variants in the architecture of the keyword extraction system which was previously proposed. The process of executing the system is to encode a word into a table and classify it into keyword or non-keyword. This section is intended to describe the keyword extraction system which is implemented in this study.

The process of collecting the sample words and encoding them into tables for implementing the keyword extraction system is illustrated in Figure 9. The keyword extraction is viewed as a binary classification which classifies a word into keyword or non-keyword. The topic-based word classification belongs to the domain independent classification where a same word is always classified identically with regardless of the domain, whereas the keyword extraction is the domain dependent classification where a same word may be classified differently depending on the domain. The words which are labeled with keyword or non-keyword are collected domain by domain. It is required to tag the input text with its own domain for executing the keyword extraction.

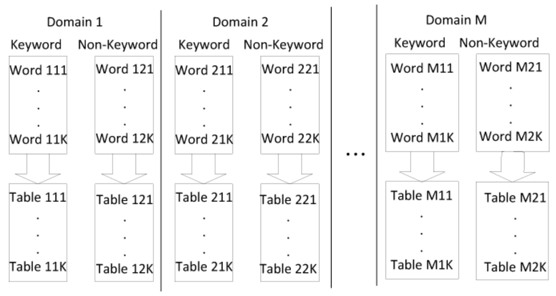

Figure 9. Preparation of Training Examples

The entire architecture of the proposed keyword extraction system illustrated in Figure 10. A text is given as the input and words are extracted from it in the indexing module. The sample words in the keyword group and the non-keyword group are encoded into tables in the encoding module. The words which are indexed from a text are classified into one of the two categories in the similarity computation module and the voting module. The difference from the system architecture which is proposed in [8] is to replace the initial version of KNN algorithm by its variants.

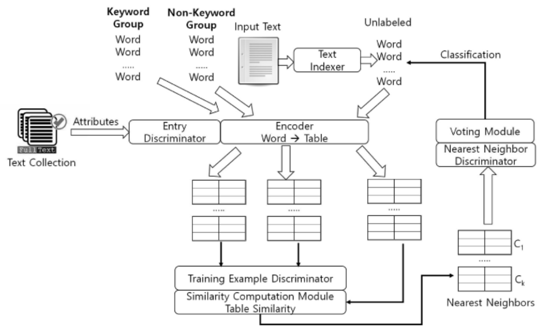

Figure 10. System Architecture

The execution process of the proposed system is illustrated as a block diagram in Figure 11. The sample words which are labeled with keyword or non-keyword are collected in each domain, and they are encoded into tables. An input text is indexed into a list of words, and they are also encoded into tables. One of the three KNN variants is selected and applied to the classification of each word into keyword or non-keyword. The words which are classified into keyword are extracted as the final output.

Let us make some remarks on the proposed system which is illustrated in Figure 10 as its architecture. The M domains are decided in advance, sample words each of which is labeled with keyword or non-keyword are prepared in each domain, and they are encoded into tables. The words which are indexed from a text are classified by one of the three KNN variants which are described in Section III-C. The difference from the previous version of keyword extraction system is to provide the selection of the nearest neighbor

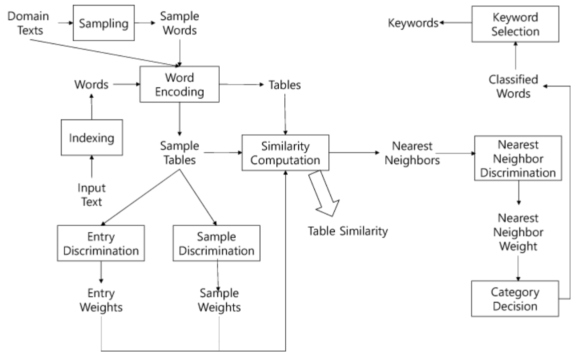

Figure 11. System Flow

discrimination, the entry discrimination, and the training example discrimination. We expect the better performance by replacing the initial version by its variants.

IV. Conclusion

Let us mention the significances of this research as the conclusion. We encode words into tables and apply the similarity metric among them to the KNN variants as well as the standard version. We derive the three variants from the standard version by discriminating the nearest neighbors, the entries, and the training examples. We design the keyword extraction system by adopt the KNN variants with the proposed similarity metric. In the next research, we will implement the keyword extraction system as a real system.

Let us mention the remaining tasks as the further discussions on this research. It is necessary to improve the speed of computing the similarity between tables in the implementation level. Other machine learning algorithms such as the Naïve Bayes and the SVM (Support Vector Machine) are modified into the table-based versions. The proposed versions of the KNN variants are applied to other tasks such as the text classification and the text summarization. The keyword extraction system should be implemented by adopting the proposed approaches as a real program.

References

[1] E.H. Han, G. Karypis and V. Kumar, "Text Categorization Using Weight Adjusted k-Nearest Neighbour Classification" pp53-64, in Advances in Knowledge Discovery and Data Mining. PAKDD 2001, Berlin, Heidelberg:Springer, 2035, 2001.

[2] T. Jo, "Table based Matching Algorithm for Soft Categorization of News Articles in Reuter 21578", 875-882, Journal of Korea Multimedia Society, Vol 11, No 6, 2008.

[3] H. Parvin, A. Hosein, and M.B. Behrouz, "MKNN: Modified k-nearest neighbor" Proceedings of the World Congress on Engineering and Computer Science. Vol. 1. Newswood Limited, 2008.

[4] T. Jo, "Device and Method for Categorizing Electronic Document Automatically", 10-2009-0041272, 10-1071495, 2011.

[5] T. Jo, "Normalized Table Matching Algorithm as Approach to Text Categorization", 839-849, Soft Computing, 19, 4, 2015.

[6] T. Jo, "Table based KNN for Extracting Keywords", 812-817, The Proceedings of 18th International Conference on Advanced Communication Technology, 2016.

[7] T. Jo, "Table based K Nearest Neighbor for Word Categorization in News Articles", The Proceedings of 25th International Conference on Computational Science & Computational Intelligence, 2018.

[8] T. Jo, "Keyword Extraction in News Articles using Table based K Nearest Neighbors", The Proceedings of 25th International Conference on Computational Science & Computational Intelligence, 2018.

[9] A.A. Nababan and O. S. Sitompul, "Attribute weighting based K-nearest neighbor using Gain Ratio", Journal of Physics: Conference Series, 1007, 1, 2018.

[10] T. Jo, "Keyword Selection from Textual Data using Table based K Nearest Neighbor", in Preparation, 2023.